

mmf 1.4

methodica message feeder

User Guide

Document Code: mmf.7602.201008.en

Table of contents

Table of contents.....	1
1 Introduction.....	3
1.1 What it is	3
1.2 How it works.....	4
2 mmf/Feeder	9
2.1 Formats supported.....	9
2.2 Embedding a feed in a web page.....	10
3 mmf/Editor.....	13
3.1 The online editor interface.....	13
3.2 Feeds.....	15
3.3 Messages.....	22
4 mmf/MPump	26
4.1 Purpose.....	26
4.2 How it works.....	26
4.3 Formatting a message e-mail.....	27
5 Appendix	29
5.1 Formatting tags	29
5.2 Variable references.....	29
5.3 Embedded feeds.....	30
5.4 Configuration files	39

Figures

Figure 1: System Architecture.....	4
Figure 2: Software Architecture.....	5
Figure 3: Data Structure.....	5

Tables

Table 1: File Structure.....	6
Table 2: Style classes	10
Table 3: The fields of a feed.....	17
Table 4: The fields of a message	23
Table 5: Formatting tags	29
Table 6: Variables	29
Table 7: Configuration files and parameters	39

Revision history

Date	Reason	Author
2010-07-02	First edition	hhe
2010-08-17	Update, corrections	hhe

Trademarks:

- UNIX is a registered trademark of The Open Group
- Linux is a registered trademark of Linus Torvalds
- Solaris is a registered trademark of Oracle Corporation
- Windows is a registered trademark of Microsoft Corporation
- methodica, methodica.ch, methodica message feeder, mmf/Editor, mmf/Setup, mmf/Backup, mmf/Report, and mmf/Sysinfo are trademarks of methodica herger, Switzerland.

Copyright:

We reserve all rights of authorship and ownership for this document and the software described herein.

Copyright © 2010, methodica herger.

1 Introduction

1.1 *What it is*

Methodica Message Feeder (MMF) is an application that supports...

- **publishing message feeds in these formats: RSS, ATOM, and HTML**, and
- **administering the feed and message data.**

The feed and message data is stored in a database (currently MySQL).

The application includes components for...

- **providing the data to readers and browsers**,
- **editing feeds and messages**, and
- **importing messages from e-mail boxes.**

All components are working web-based. Thus, you can consume the messages and edit feeds and messages...

- **without the need to install a client component**
(Standard browsers, such as Internet Explorer, Firefox, etc., or RSS / ATOM reader applications will be sufficient),
- **from anywhere in the world or within an enterprise environment**
(internet / intranet access is necessary),
- **concurrently**
(many users may consume and edit the centrally stored feed data at the same time).

New feeds can be created through the online editor interface.

Messages can be added through...

- the online editor interface, or
- via e-mail.

MMF currently is available in English and German.

Its components as well as the database support UTF-8 character sets. So, the feed and message data may include character sets like Japanese, Chinese, Russian, Greek, Thai, Hindi, Arabic, etc. MMF also supports the direction right-to-left (Arabic languages, Hebrew), although not all browsers and feed reader clients do so.

1.2 How it works

1.2.1 System Architecture

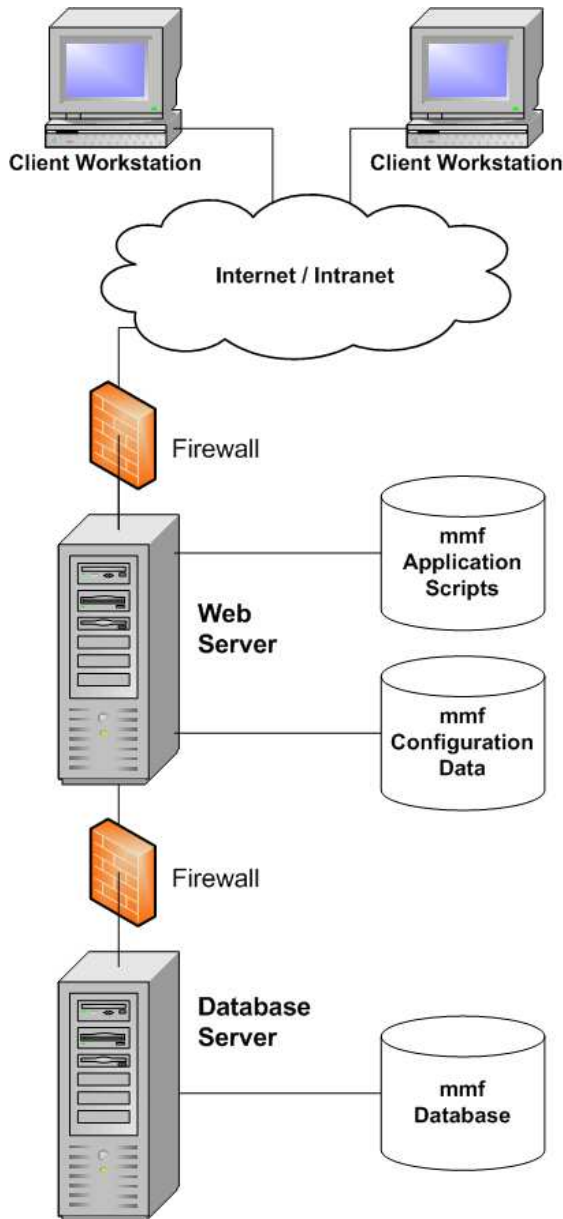


Figure 1: System Architecture

The application is integrated into a web server.

Besides the database service, all the components are web application scripts.

The web and database services may be installed on different machines or may reside on one machine.

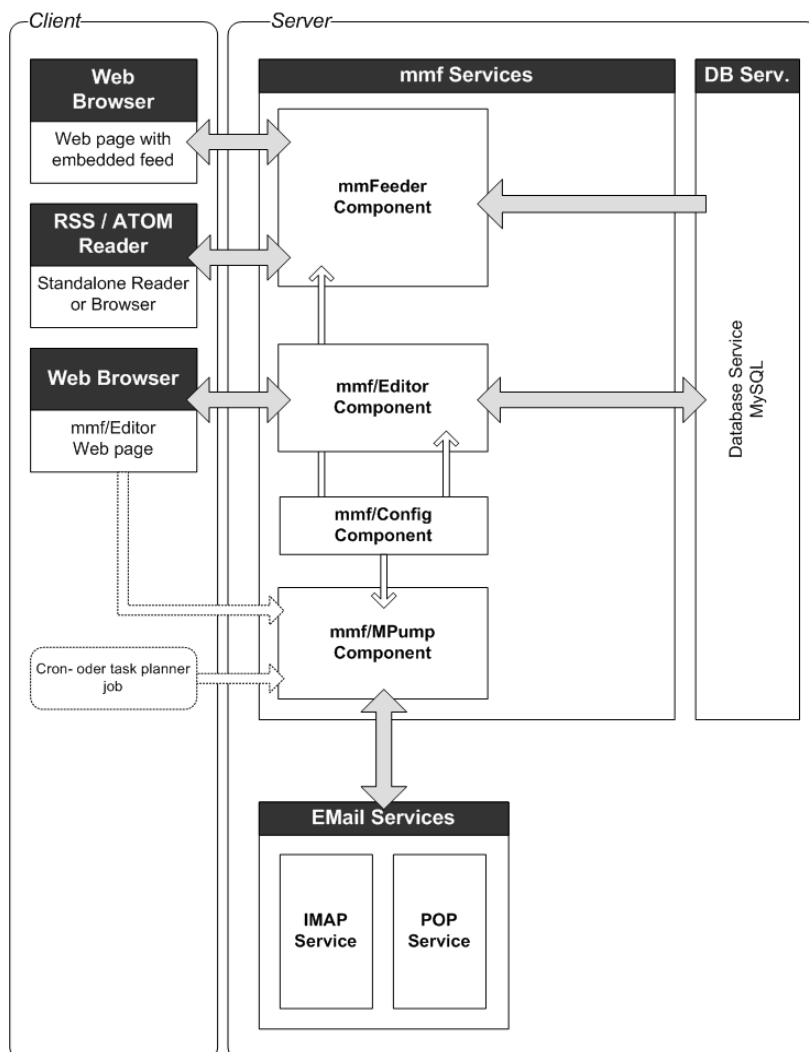
The browser installed on the client workstation, or an RSS / ATOM reader client application sends an HTTP request to the web server. The web server then will include the appropriate MMF component, which will prepare the resulting text, and sends the result back to the client.

Feed data may be requested from within any web page (embedded, web-2 technology using Ajax¹ and JavaScript). The mmf editor component creates standard web pages (HTML).

For security reasons the web servers may be protected by firewalls, and you may be asked to authenticate before accessing the (intranet) web server and the application at all.

¹ AJAX = Aynchronous JavaScript and XML

1.2.2 Software Architecture



The mmf main components:

- **mmf/Feeder**
Provides feeds to consumers in all supported formats.
- **mmf/Editor**
Allows for online editing feed and message data in a comfortable way.
- **mmf/MPump**
Imports messages from e-mail accounts into the database. mmf/MPump accesses external e-mail services. The component acts as an e-mail client.

On the client side, for all operations, a standard web browser only is necessary

Figure 2: Software Architecture

1.2.3 Data

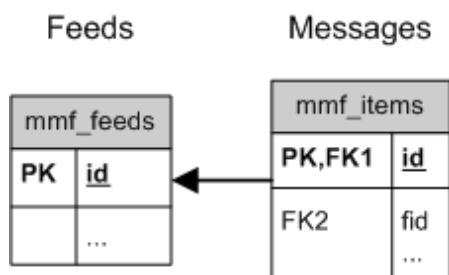


Figure 3: Data Structure

The mmFeeder data is stored in a database (currently MySQL).

The data is structured in just 2 entities:

- **Feeds**, and
- **Messages**.

Messages are assigned to feeds.

There may be any number of feeds, and any number of messages per feed, at a time.

Feeds are identified by their name, a 24-character string. Messages are linked to feeds via this name. Messages are identified by an automatically generated numeric key.

1.2.4 File Structure

The table below shows the application directories and files together with a brief description of its purposes.

The fiornat path name *path_to_mmfeed* means the drive and path down to the MMF directory , e.g.

X:\webserver\www.mydomain.tld\mmfeed or
/var/www/www.yourdomain.tld/web/mmfeed/.

Which files may be modified?

- Files that never may be modified are marked **red**,
- Those an administrator might modify carefully are marked **orange**, and
- Those an administrator may modify at his/hers will are marked **green**.

(See marks in the column titled **M**).

Notes: All modifications should be done by an administrator as they will have implications for all users.
 Carefully follow the appropriate syntax; otherwise the application might become unusable.

Table 1: File Structure

M	File	Purpose
	path_to_mmfeed	
	_config.inc	Basic configurations ² .
	_licinfo.php	Provides license information to be used by _sysinfo1.php (support only) .
	_sysinfo1.php	Provides information about the application (support only).
	_sysinfo2.php	Provides information about the server (support only).
	feed_sample.html	Samole web page with embedded feed.
	index.php	Mmf/Feeder component, Logik.
	userpar.ini	User variables for reference in feeds an messages ² .
	pfad_zu_mmfeed/admin	
	_credentials.cfg	Username and password when the <i>admin</i> sub-directory is protected by .htaccess.
	_db.cfg	Database configuration ² .

² See 5.4 "Configuration files", page 39

M	File	Purpose
	_debug.cfg	Debug configuration ² .
	_default.cfg	Default feed coniguration ² .
	admin_tpl.htm	mmf/Editor template for the user interface.
	index.php	mmf/Editor component, logic.
	de.www.mydomain.tld.cfg	Feed configuration file, default, by setup.
	en.www.mydomain.tld.cfg	Feed configuration file, default, by setup.
	mail_feed.php	mmf/MPump component, logic.
pfad_zu_mmfeed/admin/doc		
	helpdoc.inc	Online help documentation structure.
	Miscellaneous help documentation files, language-specific.	
pfad_zu_mmfeed/admin/images		
	Pictures and icons for use by mmf/Editor, language-specific.	
pfad_zu_mmfeed/images		
	Pictures and icons for use by mmf/Feeder, language-specific.	
pfad_zu_mmfeed/library		
	langstrings.inc	Internationalisation functions.
	MMF.class.php	mmf/Feeder, logic.
	MMF_editor.class.php	mmf/Editor, logic.
	MMF_feed_form.inc	mmf/Editor, template and logic feed form.
	MMF_feed_test.inc	mmf/Editor, , logic feed test (mmf/Feeder emulation) .
	MMF_form_functions.inc	mmf/Editor, common functions for handling forms.
	MMF_item_form.inc	mmf/Editor, template and logic message form.
	MMF_login_form.inc	mmf/Editor, template and logic log-in form.
	MMF_vchk_form.inc	mmf/Editor, template and logic version checker.
	MMF_xmlreader.class.php	XML parser, logic.
pfad_zu_mmfeed/library/lang		
	de.txt	Import logic for text definitions, German.
	import_lang.inc	Import logic for text definitions, general.
	de_utf8.cod	Text definitions, German (Unicode) .
	en-GB.txt	Import logic for text definitions, English.
pfad_zu_mmfeed/scripts		
	ajaxify.js ajaxify_min.js ³	Script for dynamically getting data from the server. Use for embedding feeds in your own web pages.

³ compressed variant

M	File	Purpose
	jquery-1.4.2.min.js	mmf/Editor user interface, client logic.
	jquery-1.8.2.mmf.min.js	mmf/Editor user interface, client logic.
pfad_zu_mmfeed/setup		
	Miscellaneous files for set-up process.	
pfad_zu_mmfeed/styles		
	_default.css	Default styles (used when mmf/Feeder is directly called in HTML mode).
	mmfeed_default.css	Default styles for feeds embedded in web pages.
	mmfeed_farbe.css	Colour schemes for feeds embedded in web pages.
pfad_zu_mmfeed/styles/mmf		
	Miscellaneous styles and images for use with the mmf/Editor component.	

2 mmf/Feeder

This component provides feeds to consumers. As far as for the RSS and ATOM formats requesting a feed is a standard HTTP request; the browser or feed reader knows how to handle the data⁴.

You may also use mmf/Feeder to display messages in any HTML page, e.g. as a news panel of the home page. In this case we recommend to dynamically request feeds through Ajax and JavaScript from otherwise static or also dynamically created pages.

2.1 Formats supported

2.1.1 RSS

Format

RSS (Really Simple Syndication), version 2.0

Request (URL)

`http(s)://www.yourdomain.tld/mmfeed/index.php?feedname=feed_name&type=RSS5`

2.1.2 ATOM

Format

ATOM, version 1.0

Request (URL)

`http(s)://www.yourdomain.tld/mmfeed/index.php?feedname=feed_name&type=ATOM5`

2.1.3 HTML

Format

HTML (HyperText Markup Language, the web page definition language), version 4.0.

Request (URL)

`http(s)://www.yourdomain.tld/mmfeed/index.php?feedname=feed_name&type=HTML&option=options5`

⁴ Note: the direction right-to-left of Arabic and Hebrew languages is not correctly supported by all browsers and feed reader clients.

⁵ Notes:

- Replace the formal host name *www.yourdomain.tld* by the real host name.
- Replace the formal protocol tag *http(s)* by either *http* or *https*.
- The directory path *mmfeed* may have been changed by the administrator.

Where options (separated by the pipe character '|') are one or more of:

- **inline**
No HTML code for a standalone page is generated – just the code necessary to embed the feed.
- **nohead**
No feed data is included – just the messages.

Example:

`http://www.methodica.ch/mmfeed/?feedname=www.methodica.ch_de&type=html&option=inline|nohead`

You can add the optional parameter *openmesg=nn*. This parameter defines how many messages (nn) should be shown expanded; further messages will be displayed compacted. Default value for this parameter is 3. If you set it to -1 no messages are shown compacted.

Beispiel:

`http://www.methodica.ch/mmfeed/?feedname=www.methodica.ch_de&type=html&option=inline|nohead&openmesg=3`

2.2 Embedding a feed in a web page

You can embed a feed into any web page.

Three steps are necessary to embed a feed:

1. The page must contain the scripts to request the feed and assign the resulting text to a container (DIV) within the page (feed update).
2. The HTML code must be tailored to include the container and to start the feed update script.
3. Styles are necessary to define how the elements of the feed are displayed.

A working example of such a web page can be found here:

`http(s)://www.yourdomain.tld/mmfeed/feed_sample.html5`

The easiest way to create a web page with an embedded feed is to have mmf/Editor generate the sample page for a feed and a style sheet.

2.2.1 The styles

You can modify the look of an embedded feed through class styles. Mmf/Feeder assigns a class to every part of a feed. Styles define how elements of a class are displayed.

Table 2: Style classes

Class name	Element / State
.mmf_feed_panel	Feed panel (e.g. border, background).
.mmf_feed_header	Feed title.
.mmf_feed_header_link	Hyperlinks in the feed title (if appropriate).

Class name	Element / State
.mmf_feed_header_link:hover	Same as above; when the mouse is over the element.
.mmf_feed_text	Feed text.
.mmf_feed_text_link	Hyperlinks in the feed text.
.mmf_feed_text_link:hover	Same as above; when the mouse is over the element.
.mmf_feed_footer	Feed footer (e.g. copyright text).
.mmf_feed_footer_link	Hyperlinks in the feed footer.
.mmf_feed_footer_link:hover	Same as above; when the mouse is over the element.
.mmf_item_panel	Message panel (e.g. border, background).
.mmf_item_panel_last	Last message panel.
.mmf_item_header	Message title.
.mmf_item_header_link	Hyperlinks in the message title (if appropriate).
.mmf_item_header_link:hover	Same as above; when the mouse is over the element.
.mmf_item_text	Message text.
.mmf_item_text_link	Hyperlinks in the message text.
.mmf_item_text:hover	Same as above; when the mouse is over the element.
.mmf_item_footer	Message footer, e.g. publication date, author, etc.
.mmf_item_footer_link	Hyperlinks in the message footer.
.mmf_item_footer_link:hover	Same as above; when the mouse is over the element.
.mmf_feed_text_link IMG	Images in hyperlinks within the feed text.
.mmf_item_text_link IMG	Images in hyperlinks within the message text.
.mmf_p	Paragraphs within the feed and message text.

Style definitions are stored in so called style sheet files.

Such a style sheet can be found in the file

path_to_mmfeeder/styles/mmfeed_default.css

...where *path_to_mmfeeder* means the drive (if appropriate) and directory path above the mmfeed directory.

More style sheet files, for other colour schemes, are available in the same directory.

The original contents of the default style sheet file *mmfeed_default.css* can be found here: Appendix 5.3.1 "Styles", page 30.

2.2.2 Tailoring the HTML page

First, a DIV container with the correct id must be available for the feed update script to assign the feed contents. For example:

```
<div id="news_panel">&nbsp; </div>
```

Note: the id (*news_panel*) must be equal to the corresponding parameter in the feed update script.

Second, the BODY tag of the page must include a call to start the feed update loop. For example:

```
<body onload="loadFeed();">
```

2.2.3 The scripts

A web page with an embedded feed should contain this code in the <head>...</head> part:

```
<script type="text/javascript"
src="http://www.mydomain.tld/mmfeed/scripts/ajaxify_min.js"></script>

<script type="text/javascript">
  var refreshInterval = 60 * 1000; // 60 seconds
  var feedTimer = 0;
  var ajax_obj = 0;

  function loadFeed() {
    if (feedTimer) {
      clearTimeout(feedTimer);
      feedTimer = 0;
    }
    ajax_obj = new ajax();
    ajax_obj.openRequest(
      'index.php',
      'feedname=en.www.mydomain.tld&type=html&options=inline|nohead',
      'GET',
      'news_panel'
    );
    feedTimer = setTimeout('loadFeed()',refreshInterval);
  }
</script>
```

The first script definition (<script...*ajaxify_min.js*></script>) includes a script that handles the AJAX communication and data transfer with the host. This is necessary to dynamically load parts of a web page. A more detailed description of this script can be found at Appendix 5.3.3 "Scripts", page 34.

The second script definition (<script>...</script>) automatically and repeatedly (time interval: 60 seconds) sends a request towards the web server to get the feed. The resulting text is assigned to a DIV element in the web page whose id is *news_panel*.

Note: replace the formal feed name *en.www.mydomain.tld* with the real name of the feed you want to embed.

3 mmf/Editor

The mmf/Editor component allows for creating, browsing, publishing, editing, and deleting **feeds** and **messages**.

The editor works web-based and is called when you enter an URL like this in your browser:

http://www.mydomain.tld/mmfeed/admin.

3.1 The online editor interface

The editor interface is a web page with embedded local logic. This logic inter-operates with the server-side mmf/Editor component. It uses web-2.0 techniques implemented with the help of JavaScript and Ajax.

Immediately after the editor web page has been transferred to the client a couple of requests automatically are issued. This may take a couple of seconds. As soon as the editor is ready you can begin to work with it.

Note: when working with the online interface you may experience some delays. The reason for these is the fact that the interface does issue requests towards the web server, which usually will result in one or more database accesses. The processing may take some time. So, don't immediately repeat your clicks and wait until the requests are processed (see the status information in the bottom panel of the page).

3.1.1 The page layout

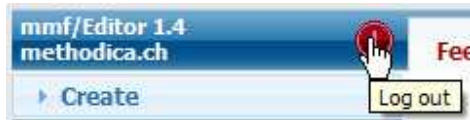


The online interface page is divided into 4 panels:

1. the **logo panel**,
2. the **menu panel**

3. the **details panel**, and
4. the **bottom panel**.

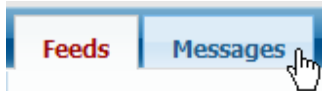
3.1.2 The logo panel



The logo panel shows the product name and version.

Additionally it contains a log-out button. If a feed is protected by a username and password pair you are asked to authenticate once you have selected the feed. When you click the log-out button all credentials you have entered before will be cleared, so no one can access these data without authenticating again.

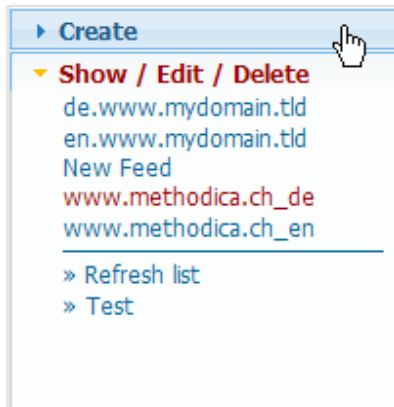
3.1.3 The details panel



On top of the details panel there are tabs to select *Feeds*, *Messages*, and *Help*. Feeds and Messages are the entities you can edit. One feed can have many messages assigned.

When you select the *Feeds* and *Messages* tabs the panel below the tabs will show the corresponding contents, and the accordion in the menu panel, plus the status information, are synchronised accordingly.

3.1.4 The menu panel



The menu panel provides an accordion whose sub-panels may be selected by clicking its caption (title bar). Each sub-panel stands for a function to apply to the data entity selected in the details panel. E.g. you can choose to create a new feed / message, or to edit or delete a feed / message.

The sub-panels and captions will be different depending on the tab that is current in the details panel.

Usually the sub-panels will contain lists of items (e.g. a list of all feeds or a list of the messages of

the current feed). Additionally they may contain action links you can click to issue an operation (e.g. to re-read a list).

3.1.5 The bottom panel

The bottom panel shows global status information plus status information for the data entity selected and the last operation applied to this data entity.

The panel contains 2 lines:

- A status message for the current data entity and operation, and
- Global information.

[Info] 0: Logged out

The status message format looks like this:

"**[class] code: text**", where...

- class = the message class (INFO, WARNING, ERROR),
- code = numeric message code, and
- text = plain text corresponding to the message code.

The background colour of the status line depends on the message class (blue, yellow, red).

0 (0) • Session open since: 2010-08-11 18:08:15 • Data updated: 2010-08-11 18:26:25

The global information line shows the copyright mark on the left side. The right side of this line is formatted like this:

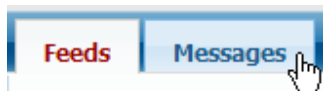
"**#requests (#queued_requests) • session_start • last_operation**", where...

- #requests = the number of outstanding requests.
A request that has been sent to the server and for which has not yet sent a response is counted.
- #queued_requests = the number of queued request that have not yet been sent to the server, but will be sent in the near future.
- session_start = date and time of the beginning of the current session. Usually this is when the editor page was last loaded.
- last_operation = date and time of the last operation performed. This may be an operation triggered by the user or an automatically triggered operation.



As long as mmf/Editor is waiting for one or more answers from the server an animated icon like the one above is displayed in the middle of the global information line.

3.2 Feeds



By selecting the corresponding tab in the details panel the editor allows for creating, editing, deleting, and testing a feed.

The accordion in the menu panel shows three selectable sub-panels:

- **Create**
The sub-panel has no active elements.
The details panel will provide a form to add a new feed.
- **Show / Edit / Delete**
The sub-panel displays a list of the names (identifiers) of all the feed already

stored. If you click on one feed name the details panel will provide an form with the details of the feed and the option to modify or delete the feed.

- **Test**

If you have selected a feed you may test it. Click the *Refresh* operation link to have the feed displayed as it would be when embedded in a web page.

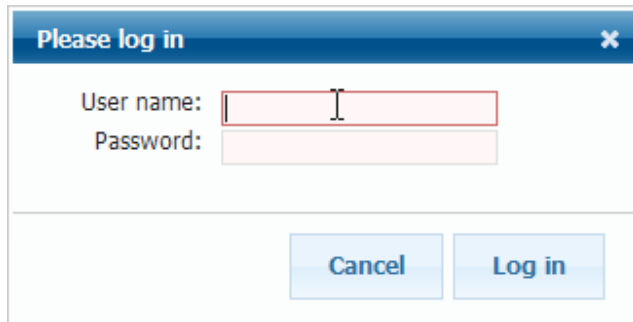
When you choose a different style sheet from the list to the right of the feed it immediately will be displayed using this style sheet.

You can also have mmf/Editor generate the code for a web page with the feed embedded by clicking *Sample page (code)* operation link.

3.2.1 User Authentication

Access may be restricted for existing feeds (access control).

Whenever you select such a feed, before the feed data will be displayed, or when you try to perform an operation on such a feed or a message assigned to such a feed, you are asked to authenticate first. A dialog window like the one shown below will be displayed:

A screenshot of a 'Please log in' dialog box. The dialog has a blue title bar with the text 'Please log in' and a close button (X). Below the title bar, there are two input fields: 'User name:' and 'Password:'. The 'User name' field is currently active, with a cursor inside. Below the input fields, there are two buttons: 'Cancel' and 'Log in'.

Enter the correct *User name* and *Password* pair for the feed, then click *Log-in*. If the credentials are ok you will be granted access to this feed and to assigned messages in the future.

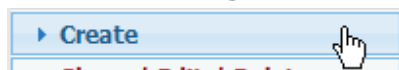
Alternatively administrators could also enter the global *MMFeed User Name* and *Password* as it was specified during the application set-up process. These are valid for all feeds.

You may be granted access to more than one feed at a time.

Once you are granted access to a feed you will no longer be prompted to authenticate again as long as the session lasts. A session ends when one of these events occur:

- you refresh the mmf/Editor page entirely,
- you open a new browser window with the mmf/Editor,
- the session has timed out,
- you willingly logged out by clicking the log-out button in the logo panel.

3.2.2 Inserting a new feed



By clicking the *Create* caption in the menu panel you will bring up a form suited to enter the feed details.

Feed [New]

Basic Parameters

Identification: (mandatory)

Title: (mandatory)

Language: (mandatory)

Description: (optional)

Extended Parameters

Link: (mandatory)

Copyright: (mandatory)

Publication time: (calendar icon)

Time-to-live: (minutes)

Maximum age: (seconds)

Image Parameters

Image title: (optional)

Image URL: (mandatory)

Link URL: (optional)

Width: (pixels)

Height: (pixels)

E-mail parameters

Host name / IP:Port: : (mandatory)

Protocol: Encryption: (mandatory)

Account: (mandatory)

☒ Validation required

Mailbox: (mandatory)

Password: (mandatory)

Feed Access Control

User name: (mandatory)

Password: (mandatory)

The fields with light-red background are mandatory and must have contents.

Table 3: The fields of a feed

Field label	Purpose	Remarks
Basic Parameters		
Identification	Feed name / identifier	Mandatory. Maximal 24 characters. Allowed: A-Za-z0-9.-_ Must be unique among all feeds stored in the database.
Title	Feed title	Obligatorisch. International character set ⁶ . Max. 64 characters.
Description	Feed outline	Optional. International character set ⁶ . Formatting tags are allowed.
Language	Feed language	A valid language code according to the ISO 639-1 standard. Format "LL-VV" (LL = language, VV = language variant).

⁶ May contain variable references.

Field label	Purpose	Remarks
Extended Parameters		
Link	Feed link	Should either be a link to read the feed or a link to a web page with more detailed information about the feed. Optional. Max. 255 characters ⁶ . Must be a valid URL.
Copyright	Copyright notice	Optional Max. 255 characters ⁶ .
Publication time	Feed publication time	May be automatically updated by mmf/Editor and mmf/MPump.
Time-to-live	Time in minutes before the feed should be re-read	Optional. Numeric. Max. 5 digits. Default: 60
Maximum age	Default publication delay	Optional. Numeric. Max. 10 digits. Default: 604800 secs = 7 days.
Image Parameters (specify if you want to include an image like a logo)		
Image title	Title of the image	Relevant only together with Image URL. Optional. Max. 64 characters ⁶ .
Image URL	URL of the feed image	Recommendation: enter a fully qualified URL here. Optional. Max. 255 characters ⁶ . Must be a valid URL. All other parameters of this group are ignored if this parameter is empty.
Link URL	Hyperlink when the image is clicked	A valid URL. Optional. Max. 255 characters ⁶ .
Width	Image width in pixels	Optional. Numeric. Max. 6 digits.
Height	Image height in pixels	Optional. Numeric.

Field label	Purpose	Remarks
		Max. 6 digits.
E-Mail Parameters (specify to import messages from mail boxes, mmf/MPump)		
Host name / IP	E-Mail host name or IP.	Optional. Max. 45 characters. e.g. imap.mydomain.tld
Port	Port corresponding to protocol and encryption.	Optional. Numeric. Max. 5 digits. Depends on the parameters <i>Protocol</i> and <i>Encryption</i> . May be different than the standard values. Standard values: <ul style="list-style-type: none"> • 143 for the IMAP proptocol • 110 for the POP proptocol
Protocol	Mail protocol	Optional. Max. 12 characters. either one of IMAP or POP. Default: IMAP.
Encryption	Encryption protocol	Optional. Max. 12 characters. SSL (Secure Sockets Layer), TLS (Transport Layer Security), NOTLS (none). Please make sure the parameter <i>Port</i> is set correspondingly. Default: NOTLS.
Validation required	Check if you want messages added by mmf/MPump to be validated by a human.	Optional. Active (checked), inactive. For security reasons we recommend to check this option. When this flag is not checked anyone could send messages messages to a mailbox which then would immediately and automatically been shown in feeds.
Mailbox	Mailbox folder	Optional. Max. 45 characters. Default: INBOX (usual).
Account	E-mail account	Optional. Max. 128 characters. Often equal to the e-mail address.

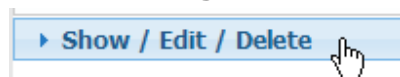
Field label	Purpose	Remarks
		The same as if you would access the e-mail account through other means.
Password	Password for accessing the account	Optional. Max. 128 characters. The same as if you would access the e-mail account through other means.
Feed Access Control (specify to protect the feed)		
User name	Name of the mmf/Editor user	Optional. Max. 64 characters. Allowed are: A-Za-z0-9 &%. -_
Password	Password for accessing the feed	Optional. Max. 64 characters. Allowed are: A-Za-z0-9 &%. -_

When you are done click the *Store* action button below the form.

If there are erroneous field contents these fields will be highlighted and the form will be presented for correction. You have the choice to either try again to store the feed with the corrected data or to *Cancel* the operation.

You might wish to *Test* the feed now. Or you could add a message to the newly created feed by selecting the *Message* tab in the details panel and click the *Create* caption in the menu panel.

3.2.3 Editing a feed



By clicking the *Show / Edit / Delete* caption in the menu panel you have the editor read the feed data from the database and present them in a form very similar to the one described under 3.3.1 "Inserting a new message", page 22. The attributes of all the fields are also identical to the ones described there.

You have these options:

- **Update**
Save the changed data in the database. The update will immediately take place and all further requests to mmf/Feeder will provide the updated data.
- **Delete**
Delete the current feed. This will also automatically delete all messages assigned to this feed. The deletion will immediately take place.
Note: this operation can not be undone!
- **Re-read**
Re-read the feed data from the database.

- **Reset**

Reverse all changes made in form since the data has been last read or updated. The feed data is not re-read from the database.

3.2.4 Deleting a feed

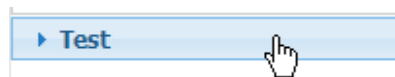
Click the *Show / Edit / Delete* caption in the menu panel. When the feed data form is displayed in the details panel you have the option to *Delete* the feed.

Notes:

- **All messages assigned to the feed are deleted too.**
- **This operation can not be undone.**

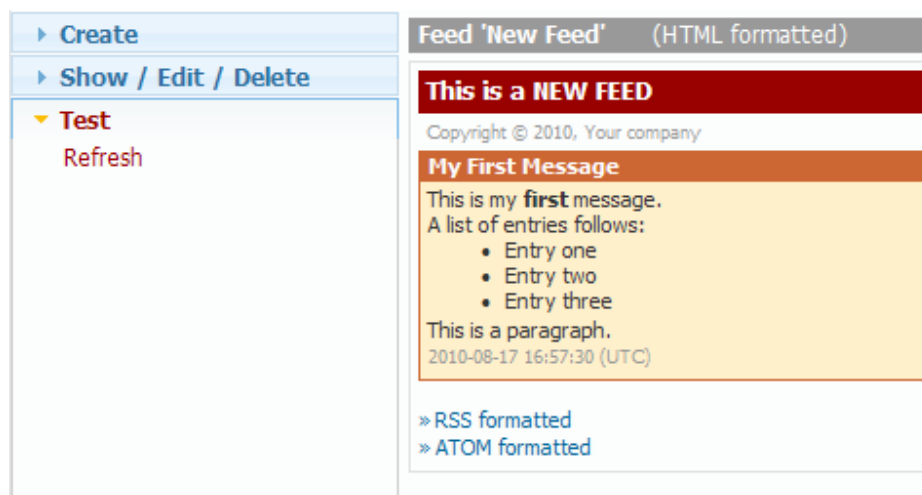
Also see 3.2.3 "Editing a feed", page 20.

3.2.5 Testing a feed



Click the *Test* caption in the menu panel. Then, in the sub-panel, click *Refresh*. This will display the feed together with its currently published messages as it would be when embedded in a web page.

The styles used are the ones written in the style sheet file *mmfeed_default.css* (see Appendix 5.3.1 "Styles", page 30). So, if you use this style sheet for embedding the feeds into your web pages the formatting will be equal.



To the right of the live feed you will find a list of predefined styles. click on one, and the look of the feed will immediately change.

You can have mmf/Editor generate the code for a sample page using the feed and style you have selected. To do so click the link *Sample page (code)* in the menu area. You then could copy the generated code and paste it in a text editor. Save it under a name like "sample.html" locally and test it in your browser. The code may well be taken as a basis for embedding a feed in your own pages.

3.2.6 Additional functions

The mmf/Editor online interface automatically starts the **mmf/MPump** component (see 4 "mmf/MPump", page 26) after the mmf/Editor online interface page has been loaded. This will be repeated periodically in 10 minutes intervals.

By mmf/MPump the messages sent to mailboxes are imported into the database and are available for releasing and editing together with the messages created through mmf/Editor.

Note: the mmf/MPump operation is performed in parallel to the mmf/Editor operation. Since mmf/Editor does not know, when mmf/MPump has imported new messages it can not refresh the list of messages. You can do this by clicking the *Refresh list* link in the left sub-panel titled *Show / Edit / Delete* when the *Mes-*
sages tab in the details panel is active.

3.3 Messages

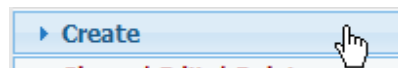


By selecting the corresponding tab in the details panel the editor allows for creating, editing, deleting a message. Of course you must before have selected a feed where the message is assigned to or where to assign a new message.

The accordion in the menu panel shows two selectable sub-panels:

- **Create**
The sub-panel has no active elements.
The details panel will provide a form to add a new feed.
- **Show / Edit / Delete**
The sub-panel displays a list of the names (identifiers) of all the feed already stored. If you click on one feed name the details panel will provide an form with the details of the feed and the option to modify or delete the feed.

3.3.1 Inserting a new message



By clicking the *Create* caption in the menu panel you will bring up a form suited to enter the message details.

Feed 'www.methodica.ch_en' :: Message [New]

Identification:

Title:

Description:

publication time: ☒ Valid / released 2010-08-23 1: - 2010-08-30 1:

Link:

Author:

Source:

Source URL:

The fields with light-red background are mandatory and must have contents.

Table 4: The fields of a message

Field label	Purpose	Remarks
Identification	Message identifier	Read only. An identification number automatically provided by mmf/Editor.
Title	Message title	Obligatorisch. Max. 64 characters ⁶ .
Description	Message outline, message text.	Mandatory. International character set ⁶ . Formatting tags are allowed (for details see Appendix 5.1 "Formatting tags", page 29). Keep as short as possible. Use the link field to provide a link to a web page with more details.
Publication time (from)	Begin of the publication period	Messages are shown when the current date / time is within the publication from ... until time frame. Format: "yyyy-mm-tt hh:mm:ss".
Publication time (until)	End of the publication period	
Link	Hyperlink to a web page with more details.	Optional. Must be a valid URL. Max. 255 characters ⁶ .
Author	Author name of the message.	Optional. Max. 64 characters ⁶ .

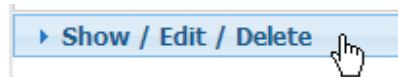
Field label	Purpose	Remarks
		Although the RSS / ATOM favour the author's e-mail address instead of its name, for security reasons, we encourage that you do enter e-mail addresses here only if you use the feed in an intranet.
Source	The message source.	Optional. Max. 64 characters ⁶ .
Source URL	A link to a corresponding web page.	Optional. Must be a valid URL. Max. 255 characters ⁶ .

When you are done click the *Store* action button below the form.

If there are erroneous field contents these fields will be highlighted and the form will be presented for correction. You have the choice to either try again to store the feed with the corrected data or to *Cancel* the operation.

You might wish to *Test* the feed with the newly created message now. To do so click the *Test* operation link in the menu panel.

3.3.2 Editing a message



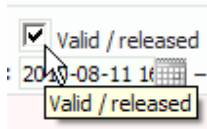
By clicking the *Show / Edit / Delete* caption in the menu panel you will be shown the list of the messages assigned to the currently selected feed. By clicking the corresponding link you ask the editor to read the message list data from the database and present it in a form very similar to the one described under 3.3.1.

"Inserting a new message", page 22. The attributes of all the fields are identical to the ones described there.

You have these options:

- **Update**
Save the changed data in the database. The update will immediately take place and all further requests to mmf/Feeder will provide the updated data.
- **Delete**
Delete the current message. The deletion will immediately take place.
Note: this operation can not be undone!
- **Re-read**
Re-read the message data from the database.
- **Reset**
Reverse all changes made in form since the data has been last read or updated. The message data is not re-read from the database.

3.3.3 Releasing a message



Messages are shown when the current date / time lies within the *publication from ... until* time frame and their *valid / released* flag is checked or the *Validation required* flag of the feed is not checked.

Messages created through the mmf/Editor component are flagged as *valid / released* while messages created through the mmf/MPump component are not.

Thus, if the feed's *Validation required* flag is checked, you also must check the message's *valid / released* flag before the message is active. This is done by an update operation (see 3.3.2 "Editing a message", page 24).

3.3.4 Deleting a message

Click the *Show / Edit / Delete* caption in the menu panel and select the corresponding message from there. When the message data form is displayed in the details panel you have the option to *Delete* the message. The deletion will immediately take place.

Note: this operation can not be undone!

Also see 3.2.3 "Editing a feed", page 20.

3.3.5 Testing a message

Since messages are delivered to a consumer (HTML page or standalone reader) in the context of a feed the feed is the entity to be tested. See 3.2.5 "Testing a feed", page 21.

3.3.6 Additional functions

There is a brief documentation available online in the editor page (*Help* tab).

There you can find

- information about the product,
- contact data, and
- a product version checker.

The version checker looks up the newest product version on one of the manufacturer's servers and compares it with the version that is installed at your site.

There is no additional data transferred together with the version check request. And no other MMF component – except mmf/MPump of course - is connecting with a server outside your site (phoning home).

Note: this feature may have been disabled by the administrator. In this case the corresponding menu option is not displayed.

4 mmf/MPump

4.1 Purpose

When a feed's E-mail parameters are correctly set the corresponding mail boxes (one per feed) are scanned by the mmf/MPump component for mail with properly formatted subject and text body.

Valid mails are interpreted and added as messages to the database.

This is another way besides the mmf/Editor that allows humans and software to add messages to a feed.

4.2 How it works

The component mmf/MPump is represented by the file

path_to_mmfeed/admin/mail_feed.php.

Whenever the mmf/Editor is used mmf/MPump is started. As long as a user has an mmf/Editor page open, this will be repeated in 10-minutes intervals.

If access to the directory *path_to_mmfeed* is password-protected (e.g by a .htaccess file) the username and password to use must be configured in the file

path_to_mmfeed/admin/_credentials.cfg.

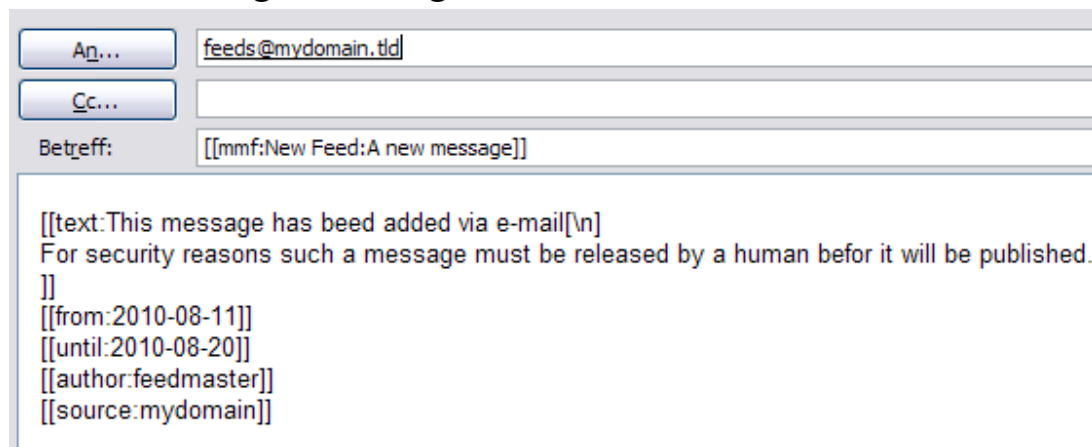
To suppress starting mmf/MPump by the mmf/Editor the constant 'SUPPRESS_MAILPUMP' in the file *path_to_mmfeed/_config.inc* to true.

Alternatively or additionally mmf/MPump may be started through CRON (Unix / Linux) or the Task Planner (Windows).

At runtime mmf/MPump, for every feed stored in the database, will scan the corresponding mail box for e-mails following the formatting rules (see below). E-mails that successfully could be interpreted and from which a message has been extracted and entered into the database will be deleted from the mail box.

You can find more about mmf/MPump in the [administrator guide](#).

4.3 Formatting a message e-mail



The screenshot shows an email client interface with the following fields:

- To:** feeds@mydomain.tld
- Cc:** (empty)
- Subject:** [[mmf:New Feed:A new message]]
- Body:**

```
[[text:This message has beed added via e-mail[\n]
For security reasons such a message must be released by a human befor it will be published.
]]
[[from:2010-08-11]]
[[until:2010-08-20]]
[[author:feedmaster]]
[[source:mydomain]]
```

For an e-mail to be honoured by mmf/MPump it must have a properly formatted subject and text body. Only e-mails with a properly formatted subject are further scanned.

The format of the subject and body text entries:

`[[tag:text]].`

The subject format:

`[[mmf:feed_name:message_title]]`

where...

- **[[mmf:...]]**
The basic sentinel without every e-mail is ignored.
Mandatory.
- **feed_name**
A valid name / identifier of the feed to assign the message to.
Mandatory.
- **Message_title**
The contents of the title field.
Optional; may be specified within the text body.

Within the text body these tags apply:

- **[[title:text]]**
The message title. Overwrites the one from the e-mail subject.
Mandatory when not specified in the e-mail subject.
Corresponds to the message field *Title*.
- **[[text:text]]**
The message text / outline.
Mandatory.
Corresponds to the message field *Description*.
Formatting tags are allowed here (see Appendix 5.1 "Formatting tags", page 29).

- **[[author:text]]**
The author name.
Optional.
Corresponds to the message field *Author*.
Alternative tags: *authorname*, *aut*.
- **[[link:url]]**
Hyperlink URL pointing to a web page with more details.
Optional.
Corresponds to the message field *Link*.
- **[[source:text]]**
Source text.
Optional.
Corresponds to the message field *Source*.
- **[[sourceurl:url]]**
Source link URL pointing to a source web page.
Optional.
Corresponds to the message field *Source URL*.
- **[[pub_from:yyyy-MM-dd hh:mm:ss]]**
Publication date / time, begin.
Optional, default is now.
Corresponds to the message field *Publication time (from)*.
Alternative tag: *from*.
Date / time format:
 - yyyy: year, 4 digits
 - MM: month of the year, 2 digits
 - dd: day of the month, 2 digits
 - hh: hour, 2 digits (24-hours a day)
 - mm: minutes, 2 digits
 - ss: seconds, 2 digits.
- **[[pub_until:yyyy-mm-dd hh:mm:ss]]**
Publication date / time, end.
Optional, default is now + *maximum age* from the feed.
Corresponds to the message field *Publication time (until)*.
Alternative tags: *until*, *to*.
Date / time format: same as for *pub_from* tag.

5 Appendix

5.1 Formatting tags

Fields, such as the description of a feed or the description of a message may contain text formatting tags. Using these tags you may set font styles (e.g. italic), font weight (e.g. bold), define the beginning of new lines, tables, lists, paragraphs, etc. The set is restricted as the feed formats RSS and ATOM formatting is restricted. And, for security reasons, not all HTML tags are supported.

Table 5: Formatting tags

Tags	Alternative(s)	Effect	Style class
[nl]	[\n], [/n]	New line	-
[tb]...[/tb]		Table, begin...end	mmf_table
[tr]...[/tr]		Table row, begin...end	mmf_tr
[th]...[/th]		Table cell within a row, begin...ind	mmf_th
[td]...[/td]		Table cell within a row, begin...ind	mmf_td
[ul]...[/ul]		Unordered llist, begin...end	mmf_ul
[ol]...[/ol]		Ordered list, begin...ind	mmf_ol
[li]...[/li]		Unordered llist, begin...end	mmf_li
[p]...[/p]		Paragraph	mmf_p
[lk:url]...[/lk]		Hyperlink with URL	mmf_item_text_link
[im:url]...[/im]		Image / picture with URL	mmf_img
[b]...[/b]		Text weight bold	-
[i]...[/i]		Text style italic	-
[u]...[/u]		Text decoration underline	-

5.2 Variable references

Where appropriate (in text fields), you may use variable references. Variable references are embraced in curly brackets {...}.

Table 6: Variables

Variable	Meaning
{server_root_url}	URL to the home page of the web site.
{date}	Date in the format "yyyy-mm-dd"
{time}	Time in the format "hh:mm:ss"
{year}	Year in the format "yyyy"
{copy}	The copyright sign ©
{httpbasepath}	URL path to mmFeeder
{filebasepath}	File system path to mmFeeder

Additionally every variable that is defined in the file:

path_to_mmfeeder/userpar.ini

...can be referenced by embracing the variable name in curly brackets. E.g.:
{company}

5.3 Embedded feeds

5.3.1 Styles

Below you find the default definition for every style class used when embedding a feed in a web page. They can be found in this style sheet file:

path_to_mmfeed/styles/mmfeed_default.css.

```
.mmf_feed_panel {
    margin:0px 0px 4px 0px; padding:0;
    color:#333333; background-color:#ffffff;
    border:1px solid #990000;
}
.mmf_feed_header {
    margin:0px 0px 0px 0px; padding:0px 4px 2px 4px;
    color:#ffffff; background-color:#990000;
    font-size:1.2em; font-weight:bold;
    cursor:pointer;
}
.mmf_feed_header_link {
    color:#336699 !important;
    font-style:italic; text-decoration:none;
}
.mmf_feed_header_link:hover {
    color:#e0f7ff !important; background-color:#999999 !important;
    text-decoration:none;
}
.mmf_feed_text {
    margin:0em 0px 2px 0px; padding:2px 4px 2px 4px;
    color:#666666;
    font-size:1.0em;
}
.mmf_feed_text_link {
    color:#666666;
    font-style:italic; text-decoration:none;
}
.mmf_feed_text_link:hover {
    color:#e0f7ff !important; background-color:#999999 !important;
    text-decoration:none;
}
.mmf_feed_footer {
    margin:2px 0px 0px 0px; padding:0px 0px 2px 0px;
    color:#999999;
    font-size:0.8em;
}
.mmf_feed_link {
    margin:2px 0px 0px 0px; padding:0px 4px 0px 0px;
    color:#336699 !important;
    font-size:0.8em;
    text-decoration:none;
}
.mmf_feed_link:hover {
    color:#333333 !important; background-color:#e0f7ff !important;
    text-decoration:none;
}
.mmf_feed_footer_link {
    color:#999999;
    text-decoration:none;
}
.mmf_feed_footer_link:hover {
```

```
        color:#e0f7ff !important; background-color:#999999 !important;
        text-decoration:none;
    }
    .mmf_item_panel {
        margin:0px 0px 4px 0px; padding:0;
        color:#333333; background-color:#fff0cc;
        border:1px solid #cc6633;
    }
    .mmf_item_panel_last {
        margin-bottom:0;
    }
    .mmf_item_header {
        margin:0px 0px 0px 0px; padding:0px 4px 2px 4px;
        color:#ffffff; background-color:#cc6633;
        font-size:1.0em;
        font-weight:bold;
        cursor:pointer;
    }
    .mmf_item_header_link {
        color:#ffffff;
        text-decoration:none;
    }
    .mmf_item_header_link:hover {
        color:#999999 !important; background-color:#e0f7ff !important;
        text-decoration:none;
    }
    .mmf_item_text {
        margin:0em 0px 2px 0px; padding:2px 4px 2px 4px;
        color:#333333;
        font-size:1.0em;
    }
    .mmf_item_text_link {
        color:336699 !important;
        font-style:italic; text-decoration:none;
        text-decoration:none;
    }
    .mmf_item_text_link:hover {
        color:#ffffff !important; background-color:#999999 !important;
        text-decoration:none;
    }
    .mmf_item_link {
        color:#333399 !important;
        font-style:italic; text-decoration:none;
        text-decoration:none;
    }
    .mmf_item_link:hover {
        color:#ffffff !important; background-color:#999999 !important;
        text-decoration:none;
    }
    .mmf_item_footer {
        margin:2px 0px 0px 0px; padding:0px 0px 2px 0px;
        color:#999999;
        font-size:0.8em;
    }
    .mmf_item_footer_link {
        color:#6699cc !important;
        text-decoration:none;
    }
    .mmf_item_footer_link:hover {
        color:#ffffff !important; background-color:#999999 !important;
        text-decoration:none;
    }
    .mmf_feed_header_link IMG,
    .mmf_feed_text_link IMG,
    .mmf_item_text_link IMG,
    .mmf_item_link IMG {
        border: 0px;
    }
    .mmf_table,
    mmf_tr,
    mmf_td {
```

```
        margin:0; padding:0; border:0;
    }
    .mmf_p {
        margin:0em 0em 0.133333em 0em;
    }
}
```

5.3.2 Sample web page

This is a complete web page with an embedded Feed.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<!--
    HINT: To test this code...
    - select it,
    - copy it to the clipboard,
    - in a text editor, create a new file and paste the code,
    - save it locally as (e.g.) 'test.html',
    - double click the file to show in the browser.
-->
<head>

<title>mmFeeder demo_de - en-GB</title>

<meta http-equiv="Content-Language" content="en-GB" />
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<!-- ##### EMBEDDING: HEADER BEGIN ##### (STEP 1) ##### -->
<link type="text/css" href="http://www.methodica.ch/mmfeed/styles/mmfeed_default.css"
rel="stylesheet" />
<script type="text/javascript"
src="http://www.methodica.ch/mmfeed/scripts/ajaxify_min.js"></script>
<script type="text/javascript">
    var nofeedhead = false; // Set to 'true' to suppress the feed header
    var refreshInterval = 5 * 60 * 1000; // Feed refresh interval (5 minutes)
    var feedTimer = 0;
    var ajaxObj = 0;

    function loadFeed() {
        if (feedTimer) {
            clearTimeout(feedTimer);
            feedTimer = 0;
        }
        ajaxObj = new ajax();
        ajaxObj.openRequest(
            'http://www.methodica.ch/mmfeed/index.php',
            'feedname=demo_de&language=en-
GB&type=html&options=inline'+((nofeedhead)?'|nohead':''),
            'GET',
            'news_panel'
        );
        feedTimer = setTimeout('loadFeed()',refreshInterval);
    }
</script>
<!-- ##### EMBEDDING: HEADER END ##### (STEP 1) ##### -->

<style type="text/css">
BODY {
    font-family:tahoma,arial,Helvetica,sans-serif;
    font-size:76%;
}

#container {
    width: 600px;
    margin: 10px auto;
    background-color: #fff;
    color: #333;
    border: 1px solid gray;
}
```

```

#header_panel {
    padding: 0.3333333em;
    background-color: #ddd;
    border-bottom: 1px solid gray;
}

#header_panel h1 {
    padding: 0;
    margin: 0;
}

#left_panel {
    float: left;
    width: 400px;
    height: 500px;
    margin: 0;
    padding: 0.3333333em;
}

#right_panel {
    height: 500px;
    margin-left: 400px;
    border-left: 1px solid gray;
    padding: 0.3333333em;
    font-size: 0.9em;
    overflow: auto;
}

#footer_panel {
    clear: both;
    margin: 0;
    padding: 0.3333333em;
    color: #333;
    background-color: #ddd;
    border-top: 1px solid gray;
}

h2 { margin: 0; padding: 0.3333333em 0 0.3333333em 0; }
</style>

</head>

<!-- ##### EMBEDDING: BODY BEGIN ##### (STEP 2) ##### -->
<body onload="loadFeed();">
<!-- ##### EMBEDDING: BODY END ##### (STEP 2) ##### -->

<div id="container">
    <div id="header_panel">
        <h1>Header</h1>
    </div>
    <div id="left_panel">
        <h2>Text Title</h2>
        <p>
            Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla dignissim ligula
            sed turpis feugiat sit amet dignissim mi vehicula.
        </p>
        <h2>Text Title</h2>
        <p>
            Proin at mi mauris. In sollicitudin tincidunt lacus, mollis ultricies elit conse-
            quat id. Vivamus commodo placerat velit, sit amet bibendum lectus convallis eget. Etiam
            tincidunt scelerisque mauris, quis consequat mauris imperdiet vel.
        </p>
    </div>
    <div id="right_panel">
        <!-- <h2>Feed Title</h2> -->
    </div>
</div>

<!-- ##### EMBEDDING: CONTAINER BEGIN ##### (STEP 3) ##### -->
<div id="news_panel">
    <!-- We cannot display the page correctly if JavaScript is not enabled. -->
    <noscript>

```

```

        <p style="margin:2.0em 0em 0em 0em; font-size:1.2em; font-weight:bold; padding:4px 4px 4px 4px; color:#990000; border:2px solid #990000;">
        Note:<br>JavaScript must be enabled for this page to function properly!<br>
        If you had JavaScript enabled you would see a message feed here.</p>
        </noscript>
    </div>
<!-- ##### EMBEDDING: CONTAINER END ##### (STEP 3) ##### -->

    </div>
    <div id="footer_panel">
        Footer
    </div>
</div>

</body>
</html>

```

5.3.3 Scripts

To dynamically update a feed from within a web page you need 2 scripts:

1. **A script to handle AJAX requests (see below).**
2. **A small script embedded in the web page itself.**

The script (2) can be found under

- 2.2.3 "The scripts", page 12, or under
- 5.3.4 "Sample page", page 37.

Das Skript (1) can be found in the file

- a) *path_to_mmfeed/scripts/ajaxify.js*
- or – in a more compact form in the file
- b) *path_to_mmfeeds/scripts/ajaxify_min.js*.

Please find the verbose variant a) of the file below:

```

/* *****
AJAXIFY.JS: A simple AJAX script

Copyright (C) 2010 methodica (http://www.methodica.ch)

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, version 3 of the License.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
The author may UNDER NO CIRCUMSTANCES BE MADE LIABLE FOR ANY
CONSEQUENCES OF USING THE PROGRAM. IF THIS WAIVER OF ALL LIABILITY
CONTRADICTS THE LAW WHERE YOU INTEND TO USE THE PROGRAM ITS USE IS
STRICTLY PROHIBITED.
See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program. If not, see <http://www.gnu.org/licenses/>.

***** */

// Ajax state values
var READY_STATE_UNINITIALIZED = 0;
var READY_STATE_LOADING = 1;
var READY_STATE_LOADED = 2;
var READY_STATE_INTERACTIVE = 3;
var READY_STATE_COMPLETE = 4;

// Ajax request counter

```

```
var OPEN_AJAX_REQUESTS = 0;

// Get element object from element id
function getElement(id) {
    return document.getElementById ? document.getElementById(id) : document.all ? document.all(id) : document.layers[id];
}

// Expand / contract containers
function togglepnl(id){
    if(!id) { return; }
    id = typeof id === "string" ? getElement(id) : id; if(!id) return;
    var expand = (id.style.display=='none');
    id.style.display=(expand ? "" : "none");
}

// Adjust element alignment and text direction for arabic and hebrew text
function adjustDir(inp,html) {
    if (!inp) return; inp = typeof inp === "string" ? getElement(inp) : inp; if (!inp) return;
    var xc_arabic = '\\u0600-\\u06FF\\u0750-\\u077F\\uFB50-\\uFDFD\\uFE70-\\uFEFF';
    var xc_hebrew = '\\u0591-\\u05A0\\u05D0-\\u05EA\\u0591-\\u05A0\\u05F0-\\u05F4\\u05C0-\\u05C4';
    var xc = new RegExp('^([0-9\\u0000-\\u0020\\u2000-\\u200D]*[ '+xc_arabic+xc_hebrew+' ])*');
    var cc = new RegExp('^([\\u0000-\\u0020\\u2000-\\u200D]*$)');
    var v = (html) ? inp.innerHTML : inp.value; if (!v) return;
    if (v.match(xc)) { // arabic character set
        inp.dir = 'rtl';
        inp.style.textAlign = 'right';
    } else if (!v.match(cc)) { // not control and punctuation character set
        inp.dir = 'ltr';
    }
}

// Ajax object
function ajax(anch) {
    this.anchname = anch;
    this.myreq = null;
    this.onerror = this.defaultError;
    this.onload = null;
    this.block = null;
    this.touttimer = null;
    this.nomsq = 0;
}

ajax.prototype.openRequest = function (url,params,httpMethod,id,onload) {
    if (!httpMethod) {
        httpMethod="POST";
    } else {
        httpMethod = httpMethod.toUpperCase();
    }
    var u = url + "?" + params;
    this.anchname = id;
    this.onload = onload;
    this.myreq = this.initXMLHttpRequest();
    if (this.myreq) {
        this.block = getElement(this.anchname);
        try {
            var loader=this;
            // Register function for readyState event
            this.myreq.onreadystatechange=function() {
                loader.onReadyState.call(loader);
            }
            this.myreq.open(httpMethod,u,true);
            var p = params;
            if (httpMethod=="POST") {
                this.myreq.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
                this.myreq.setRequestHeader("Content-length", params.length);
                this.myreq.setRequestHeader("Connection", "close");
            } else {
                this.myreq.setRequestHeader("Content-Type","text/html");
            }
        }
    }
}
```

```
    }
    // Set timeout to 30 seconds
    this.touttimer = setTimeout(function() {
        if (this.block) {this.block.innerHTML="Request timed out!";}
        this.myreq.abort();
        OPEN_AJAX_REQUESTS--;
        if (this.onload) { req.responseText=""; this.onload.call(req.responseText); }
    }, 30000);
    // Send request
    this.myreq.send(params);
} catch (err) {
    this.onerror.call(this);
}
OPEN_AJAX_REQUESTS++;
}
}

var rscript = /<script(?:\s)*?\/script>/gi;

ajax.prototype.onReadyState = function() {
    var req = this.myreq;
    var ready=req.readyState;
    var data=null;
    var coord=null;
    if (this.touttimer) clearTimeout(this.touttimer);
    this.touttimer = null;
    if ( (this.block) && (!this.nomsg) ) {this.block.innerHTML="&hellip;"; }
    if (ready==READY_STATE_COMPLETE) {
        OPEN_AJAX_REQUESTS--;
        // Extract scripts, execute, and remove them from the HTML text
        if (this.block) {
            this.block.innerHTML=req.responseText.replace(rscript, "");
            var myregexp = /<script [\s\S]*?>([\s\S]*)<\/script>/;
            var match = myregexp.exec(req.responseText);
            if (match != null) {
                try {
                    eval(match[1]);
                } catch(err) {
                    // alert(err);
                }
            }
        }
        // Execute onload() function if defined
        if (this.onload) { this.onload(this); }
    }
}

ajax.prototype.initXMLHttpRequest = function() {
    var xRequest=null;
    if (window.XMLHttpRequest) {
        xRequest = new XMLHttpRequest();
    } else if (window.ActiveXObject) {
        xRequest = new ActiveXObject ("Microsoft.XMLHTTP");
    }
    return xRequest;
}

ajax.prototype.defaultError = function() {
    if (this.touttimer) clearTimeout(this.touttimer);
    this.touttimer = null;
    OPEN_AJAX_REQUESTS--;
    alert("error fetching data!"
        + "\n\nreadyState:"+this.myreq.readyState
        + " \nstatus:      "+this.myreq.status+": "+this.myreq.statusText
        + " \nheaders:    "+this.myreq.getAllResponseHeaders()
        );
}
```

5.3.4 Sample page

This is the original sample page (HTML) code:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<!--
  HINT: To test this code...
  - select it,
  - copy it to the clipboard,
  - in a text editor, create a new file and paste the code,
  - save it locally as (e.g.) 'test.html',
  - double click the file to show in the browser.

-->
<head>

<title>mmFeeder demo_de - en-GB</title>

<meta http-equiv="Content-Language" content="en-GB" />
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<!-- ##### EMBEDDING: HEADER BEGIN ##### (STEP 1) ##### -->
<link type="text/css" href="http://www.methodica.ch/mmfeed/styles/mmfeed_default.css"
rel="stylesheet" />
<script type="text/javascript"
src="http://www.methodica.ch/mmfeed/scripts/ajaxify_min.js"></script>
<script type="text/javascript">
  var nofeedhead = false;           // Set to 'true' to suppress the feed header
  var refreshInterval = 5 * 60 * 1000; // Feed refresh interval (5 minutes)
  var feedTimer = 0;
  var ajaxObj = 0;

  function loadFeed() {
    if (feedTimer) {
      clearTimeout(feedTimer);
      feedTimer = 0;
    }
    ajaxObj = new ajax();
    ajaxObj.openRequest(
      'http://www.methodica.ch/mmfeed/index.php',
      'feedname=demo_de&language=en-
GB&type=html&options=inline'+((nofeedhead)?'|nohead':''),
      'GET',
      'news_panel'
    );
    feedTimer = setTimeout('loadFeed()',refreshInterval);
  }
</script>
<!-- ##### EMBEDDING: HEADER END ##### (STEP 1) ##### -->

<style type="text/css">
BODY {
  font-family:tahoma,arial,Helvetica,sans-serif;
  font-size:76%;
}

#container {
  width: 600px;
  margin: 10px auto;
  background-color: #fff;
  color: #333;
  border: 1px solid gray;
}

#header_panel {
  padding: 0.333333em;
  background-color: #ddd;
  border-bottom: 1px solid gray;
}
```

```

#header_panel h1 {
  padding: 0;
  margin: 0;
}

#left_panel {
  float: left;
  width: 400px;
  height: 500px;
  margin: 0;
  padding: 0.333333em;
}

#right_panel {
  height: 500px;
  margin-left: 400px;
  border-left: 1px solid gray;
  padding: 0.333333em;
  font-size: 0.9em;
  overflow: auto;
}

#footer_panel {
  clear: both;
  margin: 0;
  padding: 0.333333em;
  color: #333;
  background-color: #ddd;
  border-top: 1px solid gray;
}

h2 { margin: 0; padding: 0.333333em 0 0.333333em 0; }
</style>

</head>

<!-- ##### EMBEDDING: BODY BEGIN ##### (STEP 2) ##### -->
<body onload="loadFeed();">
<!-- ##### EMBEDDING: BODY END ##### (STEP 2) ##### -->

<div id="container">
  <div id="header_panel">
    <h1>Header</h1>
  </div>
  <div id="left_panel">
    <h2>Text Title</h2>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla dignissim ligula sed
      turpis feugiat sit amet dignissim mi vehicula.
    </p>
    <h2>Text Title</h2>
    <p>
      Proin at mi mauris. In sollicitudin tincidunt lacus, mollis ultricies elit consequat
      id. Vivamus commodo placerat velit, sit amet bibendum lectus convallis eget. Etiam tin-
      cidunt scelerisque mauris, quis consequat mauris imperdiet vel.
    </p>
  </div>
  <div id="right_panel">
    <!-- <h2>Feed Title</h2> -->
  </div>
</div>

<!-- ##### EMBEDDING: CONTAINER BEGIN ##### (STEP 3) ##### -->
<div id="news_panel">
  <!-- We cannot display the page correctly if JavaScript is not enabled. -->
  <noscript>
    <p style="margin: 2.0em 0em 0em 0em; font-size: 1.2em; font-weight: bold; padding: 4px
    4px 4px 4px; color: #990000; border: 2px solid #990000;">
      Note: <br>JavaScript must be enabled for this page to function properly! <br>
      If you had JavaScript enabled you would see a message feed here.</p>
    </noscript>
  </div>

```

```

<!-- ##### EMBEDDING: CONTAINER END ##### (STEP 3) ##### -->

</div>
<div id="footer_panel">
  Footer
</div>
</div>

</body>
</html>

```

5.4 Configuration files

The configuration files and what they define.

Parameters that should not be modified are marked **orange**, those you can modify are marked **green**.

Table 7: Configuration files and parameters

File		
M	Parameter	Purpose, default value
path_to_mmfeed/ <i>userpar.ini</i>		
	Company	Company name
	(You can add your own variables which can be referenced in feed and message fields)	
path_to_mmfeed/_ config.inc		
	SUPPRESS_VERSIONCHECK	mmf/Editor does not provide a version check link.
	SUPPRESS_MAILPUMP	mmf/Editor does not start mmf/MPump.
	(All other parameters are defined when the product has been set up)	
path_to_mmfeed/admin/_ debug.cfg		
	DEBUGFILE	Path and name of the debug file
	DEBUG	Debug mode <i>true</i> <i>false</i> , default: <i>false</i>
	DEBUGVERBOSE	Verbose debugging <i>true</i> <i>false</i> , default: <i>false</i>
	DEBUGMAXSIZE	Maximum size of the debug file. If this value is reached a new debug file will be created. Default: 307200 (= 3 KB)
path_to_mmfeed/admin/_ default.cfg		
	LANG	Language code for mmf /Editor (must be installed). Default: <i>en-GB</i> .
	TTL	Time-to-live: defines for how many minutes a feed reader should cache a feed before requesting it again.

File		
M	Parameter	Purpose, default value
		Default: 60 minutes.
	MAXAGE	Default time frame length <i>Publication time from...until</i> . Default: 604800 seconds (= 7 days).
path_to_mmfeed/admin/_credentials.cfg		
Note: Only necessary if access to the <i>admin</i> directory is protected by .htaccess.		
	username	Username, default: none
	password	Password, default: none

Notes: Changes of the configuration files should only be done by an administrator only as they have implications for all users.
Do not modify other configuration files not listed here.
Carefully follow the syntax of the modification files; otherwise the application might become unusable!